

STLSTRING

Beware of copying non-terminated strings into the STL <string> class.

Sean Barnum, Cigital, Inc. [vita¹]

Copyright © 2007 Cigital, Inc.

2007-04-17

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 3414 bytes

Attack Category	<ul style="list-style-type: none">• Denial of Service• Malicious Input		
Vulnerability Category	<ul style="list-style-type: none">• Buffer Overflow• No Null Termination		
Software Context	<ul style="list-style-type: none">• String Management		
Location			
Description	<p>Beware of copying non-terminated strings into the STL <string> class.</p> <p>The STL string class is reasonably safe, but you need to ensure that string buffer input is null terminated. If it isn't, unexpected results can occur. Look for instances where a <string> is assigned from a char buf that comes as input to a function or from outside the program.</p>		
APIs	Function Name	Comments	
	string		
Method of Attack	If an attacker can introduce an unterminated string, he or she can crash the program.		
Exception Criteria			
Solutions	Solution Applicability	Solution Description	Solution Efficacy
	When a C-style string that may or may not be terminated is used to initialize a STL string.	For input that may or may not be null terminated, replace string str = (char *) input_buf; with string str;	Effective if one knows the size of the buffer.

1. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/35-BSI.html (Barnum, Sean)

		<pre>str.append((char *) input_buf, MAX_SIZE) where "MAX_SIZE" is the maximum number of input characters you want to copy in.</pre>
Signature Details	Assignment of C-style string to STL string instance.	
Examples of Incorrect Code	<pre>/* If input_buf might not be null terminated, the following is unsafe */ string str = (char *) input_buf;</pre>	
Examples of Corrected Code	<pre>/* If input_buf might not be null terminated, the following is still safe */ string str; str.append((char *) input_buf, MAX_SIZE) /* Here "MAX_SIZE" is the maximum number of input characters you want to copy in. */</pre>	
Source Reference	<ul style="list-style-type: none"> Howard, Michael & LeBlanc, David C. <i>Writing Secure Code, 2nd ed.</i> Redmond, WA: Microsoft Press, 2002, ISBN: 0735617228. 	
Recommended Resource		
Discriminant Set	Operating Systems	<ul style="list-style-type: none"> Windows UNIX
	Language	<ul style="list-style-type: none"> C++

Cigital, Inc. Copyright

Copyright © Cigital, Inc. 2005-2007. Cigital retains copyrights to this material.

Permission to reproduce this document and to prepare derivative works from this document for internal use is granted, provided the copyright and “No Warranty” statements are included with all reproductions and derivative works.

For information regarding external or commercial use of copyrighted materials owned by Cigital, including information about “Fair Use,” contact Cigital at copyright@cigital.com¹.

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. <mailto:copyright@cigital.com>